

PinC Open Data Service

Handleiding bij de Open Data Service van provincies.incijfers.be

Inhoud

1. Algemeen
2. Technische achtergrond
3. Structuur
4. Standaardonderwerpen en metadata
5. Kubusonderwerpen
6. Aanvullende parameters
7. Gegevens ophalen met Python

Hoofdstukken 1-6 van deze handleiding zijn gebaseerd op de [handleiding van ABF Research](#) en aangepast voor provincies.incijfers.be.

In hoofdstuk 7 geven we enkele concrete programmeervoorbeelden in Python.

.

1. Algemeen

Met provincies.incijfers.be (PinC) is het mogelijk om data op verschillende manieren handmatig te exporteren. Sommige gebruikers willen echter data gestandaardiseerd en geautomatiseerd ophalen.

Daarom hebben we een Open Data Service (ODS) ingesteld op PinC. Hiermee kun je gestandaardiseerd en geautomatiseerd openbare data uit PinC ophalen via een API (Application Programming Interface).

API-key

Om de Open Data Service van PinC te kunnen gebruiken, heb je een API-key nodig, die je bij elke aanroep van de API moet meegeven. Zo'n API-key moet je eenmalig aanvragen. Contacteer hiervoor info@provincies.incijfers.be.

Oproep via URL

De Open Data Service wordt aangeroepen via een URL (Uniform Resource Locator). Binnen deze URL geef je met parameters (onderwerpen, gebieden, perioden) aan welke selectie van gegevens opgehaald moet worden. Op deze manier is een URL toekomstbestendig en zal deze bij iedere oproep de data in een identieke structuur teruggeven. De teruggave van de Open Data Service is in [JSON](#)-formaat.

Contact

Technische vragen over de Open Data Service kun je stellen via info@provincies.incijfers.be. Voor inhoudelijke vragen over de data neem je contact op met de werking [Data & Analyse](#) van je provincie.

2. Technische achtergrond

De Open Data Service maakt gebruik van OData (Open Data Protocol) v4. Dit is een internationale standaard waarmee een API volgens een vast model gedefinieerd kan worden.

Meer informatie hierover vind je op <https://www.odata.org>.

De Open Data Service van PinC vind je op <https://provincies.incijfers.be/jiveservices/odata/>. Deze URL is niet zomaar via een webbrowser benaderbaar, aangezien je in de browser geen API-key kunt opgeven. Je moet de Open Data Service dus benaderen met andere software. Je kunt een verbinding maken via bijvoorbeeld Microsoft Power BI of de ETL-tool in GIS-producten. In theorie kan het ook via Excel, maar dat is erg omslachtig en daarom gaan we daar in deze handleiding niet verder op in.

N.B.: als je de ODS toch wilt uittesten in een webbrowser, dan kun je gebruik maken van een browserextensie die je toelaat om request headers op te geven, zoals bijvoorbeeld de extensie *Modify Header Value* die beschikbaar is voor Chrome, Edge en Firefox. Je kunt hierbij eventueel ook gebruik maken van een browserextensie zoals *JSONView*, die de structuur van het JSON-object goed leesbaar maakt.

Het is daarnaast mogelijk om de Open Data Service zelfstandig te benaderen met een zelf geschreven programma. Onderstaand geven we voorbeelden in enkele gangbare programmeertalen en in een browserextensie.

Python

```
import requests
url = 'https://provincies.incijfers.be/jiveservices/odata/Variables'
headers = {'apikey': 'hier-komt-apikey'}
r = requests.get(url, headers=headers)
```

R

```
library(httr)
library(jsonlite)
url = 'https://provincies.incijfers.be/jiveservices/odata/Variables'
headers = add_headers(apikey = 'hier-komt-apikey')
r = GET(url, headers)
```

C#

```
using System.Net;
var wc = new WebClient();
var url = "https://provincies.incijfers.be/jiveservices/odata/Variables";
wc.Headers.Add("apikey", "hier-komt-apikey");
wc.DownloadString(url);
```

JavaScript

```
let url = "https://provincies.incijfers.be/jiveservices/odata/Variables";
fetch(url, {method: "GET", headers: {"apikey": "hier-komt-apikey"}})
  .then(function onRes(res) {
    if (res && res.ok) {
      // ..
    }
  });
```

Modify Header Value (browserextensie)

URL: https://provincies.incijfers.be/jiveservices/odata/
Header name: apikey
Header value: hier-komt-apikey

Daarna het vakje onder Domain leeg maken.

N.B.: vervang in bovenstaande voorbeelden de tekenreeks 'hier-komt-apikey' door je persoonlijke API-key. Vraag je eigen API-key aan via info@provincies.incijfers.be.

3. Structuur

Voor gebruik van de ODS is kennis van de datastructuur nodig. Binnen PinC worden data opgeslagen als combinatie van onderwerp, gebied en periode.

Iedere unieke combinatie van deze parameters resulteert in de data horend bij een specifiek onderwerp, voor een specifiek gebied en een specifieke periode.

De ODS gebruikt deze parameters om de gewenste data op te halen. Deze zijn onderdeel van de URL die aangeroepen wordt. Deze beschikbare parameters zijn los ook benaderbaar via de ODS.

Basis-URL

Iedere Open Data Service heeft een vaste basis-URL. Deze URL komt terug in alle aanroepen die gedaan worden. De basis-URL voor provincies.incijfers.be is

<https://provincies.incijfers.be/jiveservices/odata/>.

EntitySets

Bij het benaderen van de startpagina van de ODS zijn meerdere 'EntitySets' zichtbaar. Een EntitySet is een verzameling unieke items (Entities) die overeenkomstige kenmerken hebben. Dit zijn bijvoorbeeld eerder genoemde onderwerpen, gebieden of perioden. De meeste EntitySets zijn aanroepbaar door deze achter de basis-URL te plaatsen, bijvoorbeeld

<https://provincies.incijfers.be/jiveservices/odata/Variables>. De beschikbare EntitySets zijn:

- Variables
- GeoLevels
- GeoLevelAggregations
- GeoItems
- PeriodLevels
- Periods
- Values
- DataSources
- SwingInfo
- CubeVariables
- Dimensions
- DimLevels
- DimMembers
- DimItems
- DimMemberAggregations

Een deel van deze EntitySets is direct aan te roepen (bv. Variables), terwijl andere alleen aanroepbaar zijn in combinatie met een andere EntitySet. Een voorbeeld hiervan is GeoItems: deze kan alleen aangeroepen worden in combinatie met een gedefinieerde Entity uit GeoLevels.

Voorbeeld: [https://provincies.incijfers.be/jiveservices/odata/GeoLevels\('gemeente'\)/GeoItems](https://provincies.incijfers.be/jiveservices/odata/GeoLevels('gemeente')/GeoItems)

Wanneer een specifieke Entity opgehaald wordt, staat deze tussen haakjes en tussen enkele aanhalingstekens achter de EntitySet. In dit voorbeeld worden de items van EntitySet 'GeoItems' voor Entity 'gemeente' uit EntitySet 'GeoLevels' opgehaald. Dit geldt ook voor het ophalen van data, waarbij je een combinatie van Variables, GeoLevels, PeriodLevels en Periods moet definiëren.

4. Standaardonderwerpen en metadata

Via de ODS kunnen een aantal standaardonderwerpen en de bijbehorende metadata worden opgehaald. Hieronder volgt een toelichting van de meest gebruikte combinaties.

SwingInfo

Vraag algemene informatie over de Swing-versie op. Swing is de naam van het softwareplatform waarop provincies.incijfers.be draait.

Voorbeeld: <https://provincies.incijfers.be/jiveservices/odata/SwingInfo>

Variables

Vraag de lijst met beschikbare onderwerpen op.

Voorbeeld: <https://provincies.incijfers.be/jiveservices/odata/Variables>

Vraag de metadata van een enkel onderwerp op.

Voorbeeld: [https://provincies.incijfers.be/jiveservices/odata/Variables\('v1111a_tot_bevolking'\)](https://provincies.incijfers.be/jiveservices/odata/Variables('v1111a_tot_bevolking'))

De code van een onderwerp (in het voorbeeld hierboven: v1111a_tot_bevolking) kun je achterhalen door in de [databank](#) het gewenste onderwerp op te vragen en met de rechtermuisknop op de naam van het onderwerp te klikken. In het contextmenu kies je vervolgens 'Informatie over'. Er verschijnt een dialoogvenster, waar je op de tab 'Eigenschappen' (onder andere) de code van het onderwerp terugvindt.

DataSources

Vraag de beschikbare databronnen op.

Voorbeeld: <https://provincies.incijfers.be/jiveservices/odata/DataSources>

Vraag een enkele databron op.

Voorbeeld: [https://provincies.incijfers.be/jiveservices/odata/DataSources\('kadaster'\)](https://provincies.incijfers.be/jiveservices/odata/DataSources('kadaster'))

Het is ook mogelijk de gegevensbron van een onderwerp op te vragen.

Voorbeeld:

[https://provincies.incijfers.be/jiveservices/odata/Variables\('v1111a_tot_bevolking'\)/DataSources](https://provincies.incijfers.be/jiveservices/odata/Variables('v1111a_tot_bevolking')/DataSources)

GeoLevels

Vraag de beschikbare gebiedsniveaus op.

Voorbeeld: <https://provincies.incijfers.be/jiveservices/odata/GeoLevels>

Het is ook mogelijk de gebiedsniveaus van een onderwerp op te vragen.

Voorbeeld:

[https://provincies.incijfers.be/jiveservices/odata/Variables\('v1111a tot bevolking'\)/GeoLevels](https://provincies.incijfers.be/jiveservices/odata/Variables('v1111a tot bevolking')/GeoLevels)

ParentGeoLevels

Vraag de oudergebiedsniveaus van een gebiedsniveau op.

Voorbeeld:

[https://provincies.incijfers.be/jiveservices/odata/GeoLevels\('deelgemeente'\)/ParentGeoLevels](https://provincies.incijfers.be/jiveservices/odata/GeoLevels('deelgemeente')/ParentGeoLevels)

GeoLevelAggregations

Vraag de aggregaties van een gebiedsniveau op.

[https://provincies.incijfers.be/jiveservices/odata/GeoLevels\('provincie'\)/ParentGeoLevels\('gewest'\)/GeoLevelAggregations](https://provincies.incijfers.be/jiveservices/odata/GeoLevels('provincie')/ParentGeoLevels('gewest')/GeoLevelAggregations)

GeoItems

Vraag de beschikbare gebieden binnen een gebiedsniveau op.

Voorbeeld: [https://provincies.incijfers.be/jiveservices/odata/GeoLevels\('gemeente'\)/GeoItems](https://provincies.incijfers.be/jiveservices/odata/GeoLevels('gemeente')/GeoItems)

PeriodLevels

Vraag de beschikbare periodeniveaus op.

Voorbeeld: <https://provincies.incijfers.be/jiveservices/odata/PeriodLevels>

Het is ook mogelijk om met een onderwerp en een gebiedsniveau de beschikbare periodeniveaus op te vragen.

Voorbeeld:

[https://provincies.incijfers.be/jiveservices/odata/Variables\('v1111a tot bevolking'\)/GeoLevels\('gemeente'\)/PeriodLevels](https://provincies.incijfers.be/jiveservices/odata/Variables('v1111a tot bevolking')/GeoLevels('gemeente')/PeriodLevels)

Periods

Vraag de beschikbare perioden op.

Voorbeeld: <https://provincies.incijfers.be/jiveservices/odata/Periods>

Het is ook mogelijk om met een onderwerp, een gebiedsniveau en een periodeniveau de beschikbare perioden op te vragen.

Voorbeeld:

[https://provincies.incijfers.be/jiveservices/odata/Variables\('v1111a tot bevolking'\)/GeoLevels\('gemeente'\)/PeriodLevels\('year'\)/Periods](https://provincies.incijfers.be/jiveservices/odata/Variables('v1111a tot bevolking')/GeoLevels('gemeente')/PeriodLevels('year')/Periods)

Values

Vraag de beschikbare waarden op.

Uiteindelijk is het mogelijk om van een onderwerp met een gebiedsniveau, een periodeniveau en een periode de waarden op te vragen.

Voorbeeld:

[https://provincies.incijfers.be/jiveservices/odata/Variables\('v1111a tot bevolking'\)/GeoLevels\('gemeente'\)/PeriodLevels\('year'\)/Periods\('2016'\)/Values](https://provincies.incijfers.be/jiveservices/odata/Variables('v1111a tot bevolking')/GeoLevels('gemeente')/PeriodLevels('year')/Periods('2016')/Values)

5. Kubusonderwerpen

CubeVariables

Vraag de kubusonderwerpen op.

Voorbeeld: <https://provincies.incijfers.be/jiveservices/odata/CubeVariables>

Vraag een enkel kubusonderwerp op.

Voorbeeld: [https://provincies.incijfers.be/jiveservices/odata/CubeVariables\('kubus1112_bevolking'\)](https://provincies.incijfers.be/jiveservices/odata/CubeVariables('kubus1112_bevolking'))

DimLevels

Omdat de Open Data Service geen volledige lijst met dimensie-items op kan geven, is het mogelijk om de verschillende dimensiemembers op te vragen. Hiervoor kan je eerst een lijst met mogelijke dimensieniveaus opvragen.

Voorbeeld:

[https://provincies.incijfers.be/jiveservices/odata/CubeVariables\('kubus1112_bevolking'\)/DimLevels](https://provincies.incijfers.be/jiveservices/odata/CubeVariables('kubus1112_bevolking')/DimLevels)

DimMembers

Met een geldig dimensieniveau kan je een lijst met dimensiemembers opvragen.

Voorbeeld:

[https://provincies.incijfers.be/jiveservices/odata/CubeVariables\('kubus1112_bevolking'\)/DimLevels\('nationaliteit_3_landengroepen'\)/DimMembers](https://provincies.incijfers.be/jiveservices/odata/CubeVariables('kubus1112_bevolking')/DimLevels('nationaliteit_3_landengroepen')/DimMembers)

DimItems

Een dimensie-item is een lijst met dimensiemembers.

Dimensie-items (DimItems) zijn niet opvraagbaar maar wel in te vullen.

Welke dimensie-items er beschikbaar zijn kan je opvragen door middel van DimMembers.

Voorbeeld:

[https://provincies.incijfers.be/jiveservices/odata/CubeVariables\('kubus1112_bevolking'\)/DimItems\('nationaliteit_3_landengroepen_3'\)/GeoLevels\('gemeente'\)/PeriodLevels\('year'\)/Periods\('2016'\)/Values](https://provincies.incijfers.be/jiveservices/odata/CubeVariables('kubus1112_bevolking')/DimItems('nationaliteit_3_landengroepen_3')/GeoLevels('gemeente')/PeriodLevels('year')/Periods('2016')/Values)

Values

Uiteindelijk is het mogelijk om van een kubusonderwerp met dimensie-item, een gebiedsniveau, een periodeniveau en een periode de waarden op te vragen.

Voorbeeld:

[https://provincies.incijfers.be/jiveservices/odata/CubeVariables\('kubus1112_bevolking'\)/DimItems\('nationaliteit_3_landengroepen_1,geslacht_m'\)/GeoLevels\('provincie'\)/PeriodLevels\('year'\)/Periods\('2023'\)/Values](https://provincies.incijfers.be/jiveservices/odata/CubeVariables('kubus1112_bevolking')/DimItems('nationaliteit_3_landengroepen_1,geslacht_m')/GeoLevels('provincie')/PeriodLevels('year')/Periods('2023')/Values)

Dimensions

Vraag de beschikbare dimensies op.

Voorbeeld: <https://provincies.incijfers.be/jiveservices/odata/Dimensions>

Vraag de dimensieniveaus van een specifieke dimensie op.

Voorbeeld:

[https://provincies.incijfers.be/jiveservices/odata/Dimensions\('geboortenationaliteit'\)/DimLevels](https://provincies.incijfers.be/jiveservices/odata/Dimensions('geboortenationaliteit')/DimLevels)

Vraag de dimensiemembers van een specifiek dimensieniveau op.

Voorbeeld:

[https://provincies.incijfers.be/jiveservices/odata/Dimensions\('geboortenationaliteit'\)/DimLevels\('geboorte_nationaliteit_3_landengroepen'\)/DimMembers](https://provincies.incijfers.be/jiveservices/odata/Dimensions('geboortenationaliteit')/DimLevels('geboorte_nationaliteit_3_landengroepen')/DimMembers)

DimMemberAggregations

Vraag de aggregaties van een dimensieniveau op.

Voorbeeld:

[https://provincies.incijfers.be/jiveservices/odata/Dimensions\('geboortenationaliteit'\)/DimMemberAggregations](https://provincies.incijfers.be/jiveservices/odata/Dimensions('geboortenationaliteit')/DimMemberAggregations)

6. Aanvullende parameters

Om de data efficiënt op te halen, kan gebruik worden gemaakt van enkele aanvullende parameters. Deze worden hieronder beschreven.

Most Recent Period ('mrp')

Met de parameter 'mrp' is het mogelijk om data voor de meeste recente periode op te vragen.

Voorbeeld:

[https://provincies.incijfers.be/jiveservices/odata/Variables\('v4003_onb_perc'\)/GeoLevels\('gemeente'\)/PeriodLevels\('year'\)/Periods\('mrp'\)/Values](https://provincies.incijfers.be/jiveservices/odata/Variables('v4003_onb_perc')/GeoLevels('gemeente')/PeriodLevels('year')/Periods('mrp')/Values)

Alle mogelijkheden uit één EntitySet ('all')

Met de parameter 'all' is het mogelijk om data ongefilterd op te vragen. Deze kan toegepast worden op EntitySets GeoLevels, PeriodLevels en Periods.

Voorbeeld:

[https://provincies.incijfers.be/jiveservices/odata/Variables\('v2603_overnachting_type_hotel'\)/GeoLevels\('provincie'\)/PeriodLevels\('all'\)/Periods\('all'\)/Values](https://provincies.incijfers.be/jiveservices/odata/Variables('v2603_overnachting_type_hotel')/GeoLevels('provincie')/PeriodLevels('all')/Periods('all')/Values)

Wanneer parameter 'PeriodLevels' de waarde 'all' heeft wordt de waarde van parameter 'Periods' genegeerd. In dat geval worden waarden teruggegeven voor alle beschikbare perioden.

Uitzondering hierop is parameter 'mrp'. Bij gebruik van parameter 'mrp' bevat het resultaat per combinatie van gebiedsniveau en periodeniveau alleen data voor de meeste recente periode.

Voorbeeld:

[https://provincies.incijfers.be/jiveservices/odata/Variables\('v2712_landbouwgebruik_1'\)/GeoLevels\('all'\)/PeriodLevels\('all'\)/Periods\('mrp'\)/Values](https://provincies.incijfers.be/jiveservices/odata/Variables('v2712_landbouwgebruik_1')/GeoLevels('all')/PeriodLevels('all')/Periods('mrp')/Values)

@odata.navigationlink

Sommige EntitySets zijn onherroepelijk verbonden, zoals GeoLevels en GeoItems. Om navigatie hierin makkelijker te maken bevatten alle Entities een navigationlink naar onderliggende EntitySets. Deze is herkenbaar doordat deze start met de gerelateerde EntitySet en eindigt met @odata.navigationlink.

@odata.nextLink

Voor sommige EntitySets is een beperking ingesteld voor het aantal op te halen items. Als dit het geval is wordt automatisch een item '@odata.nextLink' toegevoegd aan de vraag. Hiermee kan een gevorderde gebruiker makkelijk door de ODS heen gaan om alle gegevens op te halen.

OData-queries

De OData v4-structuur ondersteunt een aantal standaard 'queries'. Hiermee kunnen de hierboven gedefinieerde uitvragen worden verfijnd, zodat de service geen overbodige data ophaalt. De aanvullende queries die ondersteund worden binnen de ODS zijn de volgende:

- \$filter
- \$select
- \$skip

Deze queries plaats je achter de hierboven gedefinieerde URL's. Hiermee is het bijvoorbeeld mogelijk om onderwerpen te filteren op naam, of alleen de wijken op te halen voor één specifieke gemeente, of ook om data op te halen voor meerdere (maar niet alle) perioden.

De queries beginnen altijd met een dollarteken en zijn gescheiden van de URL door een vraagteken. In de voorbeelden hieronder tonen we slechts een beperkte selectie van de mogelijkheden. Meer informatie vind je in de handleiding op <https://www.odata.org/getting-started/basic-tutorial/>.

\$filter

Voorbeeld: vraag alle databronnen op die de tekst 'Statbel' bevatten in de naam.

[https://provincies.incijfers.be/jiveservices/odata/DataSources?\\$filter=contains\(Name, 'Statbel'\)](https://provincies.incijfers.be/jiveservices/odata/DataSources?$filter=contains(Name, 'Statbel'))

Voorbeeld: vraag alle wijkdata voor het onderwerp 'private huishoudens' op, waarvan de code 'ggw7_35013' bevat. Dit zijn alle wijken binnen gemeente Oostende.

[https://provincies.incijfers.be/jiveservices/odata/Variables\('v1115a totaal huishoudens'\)/GeoLevels\('all'\)/PeriodLevels\('all'\)/Periods\('all'\)/Values?\\$filter=contains\(ExternalCode, 'ggw7_35013'\)](https://provincies.incijfers.be/jiveservices/odata/Variables('v1115a totaal huishoudens')/GeoLevels('all')/PeriodLevels('all')/Periods('all')/Values?$filter=contains(ExternalCode, 'ggw7_35013'))

Met \$filter is het ook mogelijk om data voor meerdere jaren op te vragen met één request (zonder meteen alle jaren te moeten opvragen): je kunt Period('all') opgeven en filteren op items waar de key **Period** groter of kleiner is dan of gelijk is aan de gewenste waarde(n).

Voorbeeld: vraag de gemeentelijke data voor het onderwerp 'bevolkingsdichtheid' op voor de jaren 2020 t.e.m. 2023.

[https://provincies.incijfers.be/jiveservices/odata/Variables\('vp1111a bevdicht'\)/GeoLevels\('gemeente'\)/PeriodLevels\('year'\)/Periods\('all'\)/Values?\\$filter=Period ge '2020' and Period le '2023'\)](https://provincies.incijfers.be/jiveservices/odata/Variables('vp1111a bevdicht')/GeoLevels('gemeente')/PeriodLevels('year')/Periods('all')/Values?$filter=Period ge '2020' and Period le '2023'))

\$select

Vraag alleen de velden ExternalCode en Name op.

Voorbeeld: [https://provincies.incijfers.be/jiveservices/odata/Variables?\\$select=ExternalCode,Name](https://provincies.incijfers.be/jiveservices/odata/Variables?$select=ExternalCode,Name)

\$skip

Bij sommige EntitySets wordt een beperkt aantal Entities getoond. Wanneer deze limiet bereikt is wordt aan het einde van het verzoek een @odata.nextlink geplaatst. Deze kan ook handmatig gegenereerd worden door \$skip te gebruiken.

Voorbeeld:

[https://provincies.incijfers.be/jiveservices/odata/GeoLevels\('ggw7'\)/GeoItems?\\$skip=1000](https://provincies.incijfers.be/jiveservices/odata/GeoLevels('ggw7')/GeoItems?$skip=1000)

7. Gegevens ophalen met Python

Nu je weet hoe de URL's worden opgebouwd waarmee je de Open Data Service van PinC kunt gebruiken, geven we hieronder nog enkele voorbeelden die je laten zien hoe je concreet data kunt ophalen met Python.

We maken daarbij gebruik van drie Python-modules: **requests**, **json** en **openpyxl**.

We gebruiken de module **requests** om de service te benaderen via HTTP.

Het antwoord van de service is een JSON-string. We gebruiken de module **json** om die string om te zetten in een Python-dictionary, waarna we de gegevens gemakkelijk via Python kunnen uitlezen.

Tenslotte gebruiken we de module **openpyxl** om de opgehaalde gegevens weg te schrijven naar een Excel-bestand.

Aan het begin van ons Python-script plaatsen we volgend stukje code:

```
import requests
import json
import openpyxl

odata_url = 'https://provincies.incijfers.be/jiveservices/odata/'
headers = {'apikey': 'hier-komt-apikey'}
```

De variabele **odata_url** bevat de basis-URL van de Open Data Service van PinC.

De variabele **headers** bevat de request headers. In het voorbeeld wordt één header opgegeven met als naam **apikey**. De bijbehorende waarde is de API-key. De waarde hierboven is fictief en zal niet werken. Je moet eenmalig een eigen API-key aanvragen door een mail te sturen naar info@provincies.incijfers.be.

Met de voorbeelden hieronder wordt duidelijk hoe je data kunt opvragen om verder te verwerken.

Voorbeeld 1: een lijst met de beschikbare gemeenten ophalen

Uit de voorbeelden in een eerder hoofdstuk van deze handleiding weet je dat je met deze URL de beschikbare gemeenten kunt opvragen:

[https://provincies.incijfers.be/jiveservices/odata/GeoLevels\('gemeente'\)/GeoItems](https://provincies.incijfers.be/jiveservices/odata/GeoLevels('gemeente')/GeoItems)

Als je deze URL opvraagt en het resultaat bekijkt, dan zie je dat de service een JSON-string teruggeeft die als volgt begint:

```
{
  "@odata.context":
  "https://provincies.incijfers.be/jiveservices/odata/$metadata#GeoItems",
  "value": [
```

```

{
  "@odata.type": "#Swing.ODS.GeoItem",
  "@odata.id":
  "https://provincies.incijfers.be/jiveservices/odata/GeoItems('gemeente_11001')",
  "ExternalCode": "gemeente_11001",
  "Name": "Aartselaar",
  "Description": null,
  "PointX": 151194.078125,
  "PointY": 202827.6875
},
{
  "@odata.type": "#Swing.ODS.GeoItem",
  "@odata.id":
  "https://provincies.incijfers.be/jiveservices/odata/GeoItems('gemeente_11002')",
  "ExternalCode": "gemeente_11002",
  "Name": "Antwerpen",
  "Description": null,
  "PointX": 151506.03125,
  "PointY": 217603.125
},
{
  "@odata.type": "#Swing.ODS.GeoItem",
  "@odata.id":
  "https://provincies.incijfers.be/jiveservices/odata/GeoItems('gemeente_11004')",
  "ExternalCode": "gemeente_11004",
  "Name": "Boechout",
  "Description": null,
  "PointX": 160483.90625,
  "PointY": 206966.015625
},

```

Met Python kun je deze URL als volgt opvragen:

```

json_response = requests.get(odata_url + "GeoLevels('gemeente')/GeoItems",
headers=headers)

```

De functie **requests.get()** haalt de gevraagde URL op (met de API-key in de request headers). De retourwaarde van de functie is een *response object*, dat we toewijzen aan de variabele **json_response**. Een response object bevat heel wat informatie, waaronder de statuscode (bv. 200 = oké, 404 = niet gevonden), de encoding, de headers, en natuurlijk ook de inhoud (tekst), in dit geval een JSON-string.

Die JSON-string moeten we omzetten naar een Python-dictionary om er verder mee te kunnen werken. Dat kunnen we doen met volgende Python-code:

```

geoitems = json.loads(json_response.text)

```

De JSON-string verkrijgen we uit het response object met **json_response.text**. Met de functie **json.loads()** wordt de string omgezet naar een dictionary, die we toewijzen aan de variabele **geoitems**.

De dictionary is op dezelfde manier opgebouwd als de JSON-string. We zijn geïnteresseerd in het item met als key **value**. Dat item bevat een lijst. Elk item van die lijst is op zich weer een dictionary, die de gegevens van één gemeente bevat.

Voor elke gemeente zijn er een aantal items, maar we zijn slechts in twee van die items geïnteresseerd, namelijk het item met als key **ExternalCode** en het item met als key **Name**. Die items bevatten respectievelijk de code en de naam van de gemeente.

Met een **for**-loop kunnen we alle gemeenten overlopen en telkens de code en de naam weergeven:

```
for item in geoitems['value']:
    print(item['ExternalCode'], item['Name'])
```

We kunnen ook een nieuwe dictionary maken met de code als key. Op die manier kunnen we later (zie voorbeeld 2) gemakkelijk om het even welke gemeentenaam vinden aan de hand van de code:

```
gemeenten = {}
for item in geoitems['value']:
    key = item['ExternalCode']
    val = item['Name']
    gemeenten[key] = val
```

Voorbeeld 2: een lijst met actuele waarden ophalen voor een bepaald onderwerp, voor alle gemeenten

In dit voorbeeld halen we voor een bepaald onderwerp de actuele cijfergegevens op voor alle gemeenten. We kiezen bijvoorbeeld het onderwerp met code **v1601_pand_ffgroep_6**. Dat onderwerp geeft het [aantal horecazaken](#) en is beschikbaar voor heel wat verschillende gebiedsniveaus en voor een groot aantal jaren.

Stel dat we de cijfers willen opvragen op gemeenteniveau voor het meest recente jaar (momenteel 2023), dan kunnen we daar volgende URL voor gebruiken:

[https://provincies.incijfers.be/jiveservices/odata/Variables\('v1601_pand_ffgroep_6'\)/GeoLevels\('gemeente'\)/PeriodLevels\('year'\)/Periods\('mrp'\)/Values](https://provincies.incijfers.be/jiveservices/odata/Variables('v1601_pand_ffgroep_6')/GeoLevels('gemeente')/PeriodLevels('year')/Periods('mrp')/Values)

Dat levert een JSON-string op die als volgt begint:

```
{
  "@odata.context":
  "https://provincies.incijfers.be/jiveservices/odata/$metadata#Values",
  "value": [
    {
      "@odata.type": "#Swing.ODS.Value",
      "@odata.id":
      "https://provincies.incijfers.be/jiveservices/odata/Variables('v1601_pand_ffgroep_6')/Geo
      Levels('gemeente')/PeriodLevels('year')/Values('gemeente_11001')",
      "ExternalCode": "gemeente_11001",
      "GeoLevel": "gemeente",
      "PeriodLevel": "year",
      "Period": "2023",
      "ValueString": "36"
    },
    {
```

```

        "@odata.type": "#Swing.ODS.Value",
        "@odata.id":
        "https://provincies.incijfers.be/jiveservices/odata/Variables('v1601_pand_ffgroep_6')/Geo
        Levels('gemeente')/PeriodLevels('year')/Values('gemeente_11002')",
        "ExternalCode": "gemeente_11002",
        "GeoLevel": "gemeente",
        "PeriodLevel": "year",
        "Period": "2023",
        "ValueString": "2725"
    },
    {
        "@odata.type": "#Swing.ODS.Value",
        "@odata.id":
        "https://provincies.incijfers.be/jiveservices/odata/Variables('v1601_pand_ffgroep_6')/Geo
        Levels('gemeente')/PeriodLevels('year')/Values('gemeente_11004')",
        "ExternalCode": "gemeente_11004",
        "GeoLevel": "gemeente",
        "PeriodLevel": "year",
        "Period": "2023",
        "ValueString": "30"
    },

```

Ook hier moeten we weer kijken naar het item met **value** als key. Ook hier bevat dat item een lijst. Elk item van die lijst bevat nu de gegevens voor één enkele waarde (een aantal horecazaken).

De gemeentenaam vinden we niet direct terug, maar wel de code van de gemeente (**ExternalCode**). Aan de hand van deze code kunnen we de bijbehorende gemeentenaam vinden, dankzij de dictionary **gemeenten** die we in voorbeeld 1 hebben gemaakt.

We zien dat de waarden betrekking hebben op een gemeente (**GeoLevel**) en op het jaar 2023 (**Period**), zoals we gevraagd hebben in de URL.

*N.B.: om dit voorbeeld eenvoudig te houden, hebben we slechts één jaar opgevraagd. Als je de cijfers voor meerdere jaren wil, dan kun je in de URL **Periods('all')** gebruiken in plaats van **Periods('mrp')**. Je krijgt dan alle beschikbare jaren. Het is niet mogelijk om bijvoorbeeld enkel de vijf recentste jaren op te vragen. Ben je enkel geïnteresseerd in de vijf recentste jaren, dan kun je toch alle jaren opvragen, en ervoor kiezen om enkel de cijfers van de vijf recentste jaren weg te schrijven naar een Excel-bestand.*

Met Python vragen we de URL op en zetten we de JSON-string om naar een dictionary (**values**), als volgt:

```

json_response = requests.get(odata_url +
    "Variables('v1601_pand_ffgroep_6')/GeoLevels('gemeente')/PeriodLevels('year')/Periods('mrp')/Values", headers=headers)
values = json.loads(json_response.text)

```

We willen nu een Excel-bestand creëren met vier kolommen. In de eerste kolom willen we de naam van de gemeente schrijven, in de tweede kolom de code van de gemeente, in de derde kolom het jaartal en in de vierde kolom de waarde (het aantal horecazaken).

Met onderstaande Python-code creëren we een werkblad waarop we alvast een rij met kolomtitels toevoegen:

```
wb = openpyxl.Workbook() # creëer een Excel-workbook
ws = wb.active # dit is het (eerste) werkblad
ws.append(['gemeente', 'code', 'jaar', 'aantal']) # voeg een rij toe met kolomtitels
```

Met een **for**-loop overlopen we alle items en we wijzen de gewenste waarden toe aan aparte variabelen: de code, de naam, het jaar en het aantal. De naam van de gemeente vinden we aan de hand van de code en onze eerder gemaakte dictionary **gemeenten**. Met **ws.append()** voegen we telkens een rij toe aan het Excel-werkblad:

```
for item in values['value']:
    code = item['ExternalCode']
    jaar = item['Period']
    aantal = item['ValueString']
    gemeente = gemeenten[code] # vind de gemeentenaam aan de hand van de code
    ws.append([gemeente, code, jaar, aantal])
```

Tenslotte kunnen we het Excel-bestand wegschrijven nadat we alle items overlopen hebben:

```
wb.save('horecazaken.xlsx')
```

Bovenstaande voorbeeldcode illustreert hoe je met Python data ophaalt aan de hand van de Open Data Service van provincies.incijfers.be en wegschrijft naar een Excel-bestand. Maar hier valt nog wel een en ander aan te verbeteren.

Om te beginnen staan de gemeenten in ons Excel-bestand niet netjes alfabetisch, maar in de volgorde zoals ze door de ODS aangeboden worden, d.w.z. gesorteerd op de code en niet op de naam van de gemeente. Wil je ze netjes alfabetisch, dan zul je de records eerst moeten sorteren alvorens je ze toevoegt aan het Excel-werkblad.

Houd er ook rekening mee dat de opgehaalde waarden geen getallen, maar tekenreeksen zijn. Wil je er in Excel mee kunnen rekenen, dan zul je ze eerst moeten omzetten naar getallen, bijvoorbeeld met volgende code:

```
try:
    aantal = int(item['ValueString'])
except:
    aantal = item['ValueString']
```

We gebruiken **try ... except** omdat er geen garantie is dat de waarden die de ODS teruggeeft, numeriek zijn. De waarde van **ValueString** kan bijvoorbeeld ook een streepje, een vraagteken of een x zijn voor ontbrekende, onbekende of gecensureerde aantallen. In die gevallen zou **int()** een foutmelding opleveren.

Tenslotte is het wellicht handig om de eerste rij en de eerste kolom van het Excel-werkblad te bevriezen, zodat ze in beeld blijven wanneer je doorheen het bestand scrolt. Dat kan als volgt:

```
ws.freeze_panes = 'B2'
```

De **openpyxl**-module heeft nog meer mogelijkheden om het Excel-bestand op te maken, maar deze vallen buiten het bestek van deze handleiding. Tot de mogelijkheden behoren onder andere:

de rij met de kolomkoppen in vetjes zetten, het werkblad een naam geven, de kolombreedten aanpassen, etc.

Data opvragen voor meerdere onderwerpen, gebiedsniveaus en perioden

In bovenstaande voorbeelden hebben we data opgehaald voor één bepaald onderwerp, voor één bepaald gebiedsniveau, en voor één bepaald jaar.

In de praktijk zul je wellicht een Python-script willen maken waarmee je gemakkelijk meerdere onderwerpen kunt opvragen, voor meerdere gebiedsniveaus en/of voor meerdere jaren.

Als je meerdere jaren wilt opvragen, dan kun je dat nog altijd doen aan de hand van één enkele URL (waarbij je `Periods('mrp')` vervangt door `Periods('all')`). Wil je echter data voor meerdere onderwerpen of voor meerdere gebiedsniveaus opvragen, dan heb je daar telkens een andere URL voor nodig: met één URL kun je immers maar data voor één onderwerp en voor één gebiedsniveau ophalen.

In de voorbeelden hierboven hebben we het onderwerp, het gebiedsniveau en de periode hard gecodeerd in de URL. In een algemener Python-script kun je daarvoor beter variabelen gebruiken, bv.:

```
var_code = 'v1601_pand_ffgroep_6'
geolevel_code = 'gemeente'
periodlevel_code = 'year'
period_code = 'all'

json_response = requests.get(odata_url +
f"Variables('{var_code}')/GeoLevels('{geolevel_code}')/PeriodLevels('{periodlevel_code}')/Periods('{period_code}')/Values", headers=headers)
```

Je kunt dan werken met een lijst die meerdere onderwerpcodes bevat, waarbij je in een **for**-loop telkens een andere onderwerpcodes toewijst aan de variabele **var_code**, en/of met een lijst van gebiedsniveaus, waarbij je telkens een andere gebiedscode toewijst aan **geolevel_code**.

Wellicht wil je dan ook eerst alle data van meerdere requests verzamelen en combineren alvorens je de data toevoegt aan een Excel-werkblad. Maar aangezien dit een handleiding ODS is en geen cursus Python, gaan we hier niet verder op in. Met bovenstaande voorbeelden en een basiskennis Python kom je vast al een heel eind!

Foutmeldingen

Het kan natuurlijk gebeuren dat er iets fout gaat. Wanneer je de URL niet correct opgebouwd hebt, dan zal de ODS niet de verwachte data, maar een foutmelding teruggeven.

Probeer je bijvoorbeeld data op te vragen voor een niet-bestaande onderwerpcodes, dan zul je een JSON-string terugkrijgen die geen **value**-key, maar wel een **error**-key bevat, bv.:

```
{
  "error": {
    "code": "",
    "message": "Variable with key foute_onderwerpcode not found"
  }
}
```

Voordat je verder gaat met het verwerken van de JSON-string controleer je daarom beter eerst of je geen foutmelding hebt teruggekregen, bv.:

```
json_response = requests.get(odata_url +
"Variables('foute_onderwerpcode')/GeoLevels('gemeente')/PeriodLevels('year')/Periods('mrp')/Values", headers=headers)
values = json.loads(json_response.text)

if 'value' in values:
    print('Alles oké!')
    for item in values['value']:
        print(item['ExternalCode'])
else:
    print('Er heeft zich een fout voorgedaan.')
    if 'error' in values:
        print(values['error']['message'])
```

Los van de Open Data Service kan er ook altijd iets fout gaan wanneer je om het even welke URL opvraagt met **requests**. Daarom kijk je best ook naar de statuscode die je terugkrijgt als onderdeel van het response object.

In de voorbeelden hierboven hebben we enkel de **text** property van dat object gebruikt om de JSON-string op te vragen; met de **status_code** property krijg je de statuscode (bv. 200 betekent oké, 404 betekent niet gevonden). Een foutief opgebouwde ODS-URL kan aanleiding geven tot een 404-error. Of in zeldzame gevallen kan er zich een server-fout voordoen, in welk geval je misschien een 500- of 503-foutmelding krijgt. Daarom kun je beter na elk request de statuscode controleren. Of je kunt de property **ok** testen, die **True** oplevert wanneer de statuscode kleiner is dan 400 en **False** wanneer de statuscode 400 of hoger is:

```
json_response = requests.get(odata_url + "GeoLevels('gemeente')/GeoItems",
headers=headers)
if json_response.ok:
    geoitems = json.loads(json_response.text)
else:
    print('Er is iets fout gegaan: kan de lijst met geoitems niet ophalen. Status:',
json_response.status_code)
```

Ook wanneer de API-key ontbreekt of ongeldig is, krijg je een foutmelding (dit gaat gepaard met een 404-statuscode):

```
{
  "error": {
    "code": "",
    "message": "Guest user group not found, No access!"
  }
}
```